A Benchtop Instrument Simulator for CERES/EOS-AM1

John J. Chapman
NASA Langley Research Center
21 Langley Blvd. MS 423
Hampton, VA. 23681, USA

Abstract

The first CERES instrument was launched on NASDA's Tropical Rainfall Measurement Mission (TRMM) platform Nov. 28, 97. More CERES instruments are planned for launch on NASA's Earth Observing System (EOS) satellites. CERES measures top of the atmosphere radiative fluxes using microprocessor controlled scanning radiometers. The CERES EOS-AM1 Instrument Simulator will serve as a testbed for the testing of custom instrument commands intended to solve in-flight anomalies of the instruments which could arise during the CERES EOS-AM1 mission. The concept of using an identical but non-flight qualified microprocessor and electronics ensemble linked to a virtual instrument with identical system software results in a relatively inexpensive simulation system capable of high fidelity. The first CERES Instrument Simulator, resembles the TRMM CERES instrument sufficiently well to allow benchtop testing and functional verification of microprocessor loads for TRMM instrument uploads, accurate right down to the actual checksum. Each version of the CERES Instrument Simulator consists of electronic circuitry identical to the flight unit's twin microprocessors and telemetry interface to the supporting spacecraft electronics and two personal computers (PC) connected to the I/O ports that control azimuth and elevation gimbals. Flight simulation software consists of the unmodified TRW developed Flight Code and Ground Support Software specific to the platform under study which serves as the instrument monitor and also links NASA/ TRW developed engineering models of the gimballed instrument. The CERES engineering development software models were modified to provide a virtual instrument running in real-time on a second PC linked to the flight microprocessors instrument control ports. The cost of the electronics and development of such a simulation system is dwarfed by the cost to develop the actual flight software, which has already been invested in building the actual instrument. Thus for the simulator, flight software re-use is very cost effective. The concept of using dedicated electronics and specific flight software matched to the instrument being studied allows for high fidelity simulations of each instrument system. The differences in flight code that represent the final flight configurations from platform to platform are accurately modeled. The overall task of tailoring the simulation system hardware to a given instrument platform reduces to the art of interfacing and programming a pair of embedded microprocessors linked to a pair of commercially available PCs.

Key Words: Software/Simulation/Satellite/Microprocessor/Virtual/Instrument

1. INTRODUCTION

The Clouds and the Earth's Radiant Energy System (CERES) instruments were built by TRW, Redondo Beach, CA. for NASA, Langley Research Center. The first launched CERES instrument is on the Tropical Rainfall Measurement Mission (TRMM) satellite. It is now orbiting earth (See Ref. 1 & 3). Four more CERES instruments will be launched on NASA's Earth Observing System (EOS) satellites, two on EOS AM-1, which is scheduled for launch in June 1998 and two on EOS PM-1 scheduled for December, 2000. The CERES instrument measures top of the atmosphere radiative fluxes (See Ref. 2) using microprocessor controlled gimballed radiometers. The standard commanding of each instrument is done by the Flight Operations Team (fig.1) using CERES instrument functional modes chosen for the daily operational scenarios predetermined by the Mission Operations and Project Management and Science Teams. The CERES commands consist of mnemonics grouped in long or short command formats or sequences of commands which modify the instrument behavior, for example the elevation scan limits may be adjusted for sun avoidance during biaxial scanning profiles, or special instrument calibrations or diagnostics. The engineering prototype of the instrument, normally used for ground based testing after the flight production models are launched was incorporated into the TRMM spacecraft, thus saving the expense to build one of the flight models. This decision left the Flight Operations Team without a testbed instrument for the ground based investigation of potential instrument anomalies. The CERES instrument simulator allows testing of memory patches, custom commands or instrument command sequences prior to uploading by telemetry link to the orbiting CERES instrument.

2. SIMULATOR ARCHITECTURE

Each simulator CPU consists of an 80C186 microprocessor on its own circuit card loaded with the TRW flight software stored in EPROM. Direct memory access (DMA) between the microprocessors and shared RAM for the telemetry linkage is used to communicate with the spacecraft or Bench Checkout Unit (BCU). The Ground Support Equipment (GSE) software running on the BCU dedicated PC serves as the instrument monitor. The BCU software was originally developed at TRW to provide a housekeeping screen of the telemetry parameters being monitored by the CERES instrument microprocessors while under test. Test instrument commands to be verified by the simulator are loaded into the GSE/BCU PC from the Command Maintenance Utility (CMU), then sent to the ICP via the telemetry link (which is accomplished by a coaxial cable for the simulator) to the Spacecraft Interface card. The GSE/BCU housekeeping screen displays the command status, the current and previous instrument mode (fig.3), elevation and azimuth gimbal data, plus the microprocessor memory checksum data of the uploaded patch. Since the simulator has no real gimballed components, a second PC interfaced to the appropriate microprocessor control ports provides a separate virtual azimuth and elevation gimbal. Engineering software models of the elevation and azimuth gimbals have been modified to provide a fast, virtual instrument running in real time, interrupt linked to both the ICP and DAP azimuth & elevation control ports.

3. SIMULATOR HARDWARE vs. FLIGHT CIRCUITRY

A set of bare circuit cards identical to the flight design was obtained from the flight item vendor to ensure a functionally identical electronic platform (See Ref. 4). Commercial grade integrated circuits, in plastic packages but equivalent in function to the flight parts were installed in sockets except for the telemetry handler chipset. Space qualified integrated circuits used for the flight articles are fabricated with radiation hardened components and then environmentally tested, they are expensive for a ground based simulator but must be used if no commercially equivalent part exists.

The N80C186-25MHz microprocessor chips used in the simulator are manufactured in plastic leadless chip carriers (PLCC) which fit into custom emulation sockets. The custom sockets provide for an external logic analyzer to be connected as a monitor during testing. Memory chips designed for use in PC static random access memory (SRAM) 32K X 8 and Erasable Programmable Read Only Memory (EPROM) 8K X 8 with device architecture, logic polarity and access time to match the flight components were used on the CPU memory adapter boards.

The simulator CPU cards consist of the Instrument Control Processor (ICP), Data Acquisition Processor (DAP) with two Digital Interface cards and one Spacecraft Interface card. The ICP to DAP data and address interconnect lines are provided by inexpensive ribbon cables. Other card to card wiring was made short to maintain good switching signal profiles.

The primary difference between the EOS-AM1 and TRMM simulation hardware is in the Spacecraft Interface card which has two telemetry streams, one for each of the two CERES instruments on the EOS-AM1 platform. This difference was significant enough to warrant a second simulator for the EOS-AM1 system. The second simulation system will provide for functional testing of EOS-AM1 uploads and also avoids potential changeover difficulties associated with hardware plug-in and software load mismatching. The EOS-AM1 and TRMM simulations will execute on seperate platforms.

The unmodified flight code is programmed in several EPROM chips installed on small memory adaptor boards mounted above the ICP and DAP cards, hence each simulator CPU has a memory map identical to the flight memory system. Other modifications include a manual reset switch combined with a watchdog time-out inhibit function. A non-flight grade fiberoptic to TTL converter feeds the TRMM interface between the Spacecraft Interface Card and the GSE/BCU.

The remaining CERES sensors which comprise an additional 218 channels of instrumentation such as, covers, azimuth brake, solar avoidance and component temperatures are synthesized by preloaded PC memory lookup files set to nominal levels. They are directly interfaced to the ICP and the DAP instead of the A/D converter cards. Both the ICP and DAP continually check sensor status (such as the azimuth brake) and react to these inputs accordingly to protect the instrument.

4. SIMULATOR RUNS FLIGHT SOFTWARE

The baseline flight code for each CERES instrument was originally developed by TRW for each processor. The initialization and main loop portion of the flight code is written in C++. It consists of a series of callable subroutines which are recognized by the analyzer in the subroutine display mode. The baseline microprocessor flight code was programmed into 27C64 EPROMS and the checksums were verified prior to the boot-up trials. The CERES Flight Software documentation and notes from a CERES flight software workshop were used to identify major milestones in the ICP and DAP boot-up and initialization cycles. A multichannel logic analyzer has been connected sequentially to each processor for the purpose of troubleshooting. Debugger/disassembler software has been used in conjunction with the analyzer which displays the called subroutines on the analyzer screen. Unique address/data byte pairs corresponding to the boot-up milestones were observed to verify the extent of the code execution. A fast 4 channel digital storage oscilloscope was also used to monitor certain flight code activated lines such as the memory and peripheral chip select lines to help debugging and to verify memory and peripheral chip function. The analyzer selection menu allows filtering the sampled data sets to provide displays in either hardware mode, which shows instruction fetches, memory read/write and I/O operations, software mode (which shows the machine instruction mnemonics), or subroutine call mode (which shows only the subroutine calls and the physical address). The boot-up portion of the flight code consists entirely of 80C186 assembly language, for which the hardware and software debugging screens are well suited. Once both processors are in their main control loops their subroutine calls (see Tables 1 and 2) are synchronized within a 10 msec sample window midpoint. Thereafter both loops are repeated simultaneously every 10 msec. For the DAP, the important calls used for the simulator include the Acquire Elevation Data subroutine, which reads the pseudo-encoder output and the Elevation Scan Control subroutine which sends rate and position commands to the elevation control port based on data from the scan tables in SRAM. The ICP similarly controls the azimuth control port with the Acquire Azimuth Data and Azimuth Control subroutines. The ICP also controls the telemetry interface port feeding the GSE/BCU. The logic analyzer and the BCU display provide the means to verify that the ICP and DAP do indeed communicate and properly execute the CERES flight code at the 16 MHz clock speed just as the flight instrument. The TRMM version of the CERES Instrument Simulator had only one instrument to control whereas the EOS-AM-1 platform will have two instruments designated FORE (CEF) and AFT (CEA). Since the simulation system hardware is identical for both FORE and AFT either the CEF or CEA instrument specific code can be uploaded at any given time. Thus only one EOS-AM1 instrument simulation can be performed at a given time. The procedure is to upload the flight code specific to the desired instrument and conduct the simulation.

5. VIRTUAL INSTRUMENT MODELS & LINKAGE TO MAIN LOOP

The Matlab/Simulink environment was used for modelling the instruments dynamic systems. The block diagram of the servo motor model (See fig. 4), was originally developed by engineering teams at NASA LaRC and TRW to design the CERES instruments. They accurately represent the actual system components under simulation. Minor simulator-specific modifications were made to convert the engineering analysis models into fast, real-time executable virtual instruments using the Matlab Real Time Workshop. The resultant model has been developed to be indistinguishable from the mechanical scanners to either the ICP or DAP. Both Azimuth and Elevation real-time executable models run on a 200 MHz 80686 PC and interact with the simulator microprocessor I/O port through interface cards The interface card provides a bi-directional parallel transfer of the rate, direction and position, interrupt driven by the microprocessor. An interrupt service routine was developed for processing strobed input/output (I/O) via the Metrabyte I/O card. A sample Elevation scan profile table is given in Table 3, and plotted in Fig. 5. Each inflection point in the scan corresponds to new rate, direction and position commands. Fig. 5A shows a short scan profile. In the event that the error between commanded position and actual position exceeds 500 counts for a duration of at least 100 milliseconds, the instrument will be placed in the STOW mode and the motor drive disabled. The azimuth gimbal scan profile shown in fig. 6 runs on Matlab/Simulink real-time executable code similar to the elevation model, the difference being the range, (350 deg) rate, (4 to 6 deg/sec) and specific values for friction, torque and gain. This unique feature allows the investigation of scanning commands in a virtual instrument simulation scenario without costly mechanical servo hardware or servo driver circuits in the simulator. The adjustment of the Matlab input file parameters such as drag or servo driver gain allows for modification of servo performance to reflect instrument related behavior such as the effect of mechanical wear or long term radiation effects on electronic components.

6. SIMULATOR UTILITY

The Ceres Instrument Simulator serves as a testbed for custom commands developed on the CMU. These consist of scan profiles, internal sequences, memory uploads or patches tailored to solve particular problems. Some specific examples tested on the TRMM simulator have been an ALONGTRACK profile and a special SYMMETRIC SHORT SCAN (Fig. 5B) profile, both have been prepared on the CMU. The custom instrument sequence consists of 20 consecutive cycles of standard SHORT SCAN profile followed by the SYMMETRIC SHORT SCAN. The purpose of this test is to determine if detectable radiometric offset differences exist in the space looks that would warrant either staying with the standard SHORT SCAN profile or using the proposed SYMMETRIC SHORT SCAN. The latter would provide more data taken at the +/- 63 degree per second scan rate since the rapid retrace portion of the standard SHORT SCAN would be eliminated. (See Ref. 5)

Another useful built in feature of the simulator is the BCU checksum display of ICP and DAP ram checksums. This was used during the early orbit checkout for TRMM to verify the upload status of the flight instrument. Each uploaded memory patch (there are seven operational patches to date for TRMM) alters the telemetry RAM checksum display.

7. CONCLUSION

The CERES Instrument Simulator achieves the CERES orbital instrument nominal operational states (See Fig. 7) using stored command sequence tables as accessed by internal sequence commands. Specific commands for the elevation gimbal and azimuth gimbal may be invoked to modify the instrument configurations. Uploads may further modify the allowed operational states.

The BCU screen monitors, displays and archives the important parameters such as the current and previous mode of operation, recent commands processed and ICP and DAP memory ROM and RAM checksums. The Matlab/Simulink real-time utility allows the definition and analysis of servo drive models. The executable versions provide an interrupt driven virtual mechanical instrument. The elevation and azimuth position and gimbal status data are displayed in real-time on the monitor. This provides for benchtop testing of new elevation & azimuth scan profiles, instrument scan sequences, inflection point positions, direction, torque, servo error in counts & degrees, stall and drive enabled/disabled status.

CERES test commands may be up-loaded into memory and the telemetry monitored parameters may be recorded on peripheral drive (ZIP) disks which can be played back for performance verification and then transferred to the operations team. Specific parameter plots can also be obtained from the system. Files from the logic analyzer can also be saved and transferred on disquettes.

Software modifications in the form of memory patches may thus be tested prior to uploading to measure the impact on both ICP & DAP functioning. The simulator is also useful as a training aid to provide operational familiarity to instrument technicians and engineers.

Table 1: EOS-AM1 Main Loop Calls by Instrument Control Processor

Synchronize to 100 Hz Clock
Acquire A/D Converter Data
Set Sub-mux Address for next sample
Acquire Azimuth Position and Error Values
Perform Azimuth Control Function (includes brake)
Check Spacecraft Safe-Hold Signals
Perform Solar Avoidance Check
Check for Telemetry Messages
Process Telemetry Commands
Process Current Internal Sequence
Wait until 100 Hz Sample Period Midpoint
Communicate with DAP (DMA with Shared Memory)
Perform all remaining EOS-AM Interface Tasks
Calculate ICP Code Checksums
Calculation of ICP Min./Max. Execution Times
Strobe Watchdog Timer

Table 2: EOS-AM1 Main Loop Calls for Data Acquisition Processor

Synchronize to 100 Hz Sample Clock
Acquire A/D Converter Data
Set Sub-mux Address for next sample
Read Elevation Position and Error Values
Execute Elevation Scan Profile
Perform Cover Control
Perform Closed Loop Control of Detector & Blackbody Temperatures
Perform Bolometer Bridge Balance
Update SWICS Intensity Port if necessary
Wait until 100 Hz Sample Period Midpoint
Communicate with ICP (DMA with Shared Memory)
Calculate DAP Code Checksums
Calculate DAP Min./Max. Execution Times
Strobe Watchdog Timer

Table 3 Elevation Scan Profile

Inflection Point	Sample Number	Starting Position Command	Rate & Direction Command
0	0	3237	0 deg/sec
1	50	3237	63 deg/sec
2	279	29532	0 deg/sec
3	305	29532	250 deg/sec
4	317	35317	0 deg/sec
5	342	35317	-250 deg/sec
6	354	29532	0 deg/sec
7	380	29532	-63 deg/sec
8	609	3237	0 deg/sec
9	659	3237	0 deg/sec

Acknowledgments

Special thanks to the following people for their important technical assistance:
Ernest C. Burt III of NASA, LaRC
Tom Evert of TRW Corp.
James L. Donaldson of Computer Sciences Corp.
Michael C. Holloman of NASA, LaRC
James B. Miller of NASA, LaRC
Bryant Douglas Taylor of NYMA Corp.
Paul Sakaguchi of TRW Corp.
Christopher J. Slominski of Computer Sciences Corp.

References

1. Bulletin of the American Meteorological Society, Vol. 77, No.5, May 1996
2. Bulletin of the American Meteorological Society, Vol. 76, No.11 Nov. 1995
3. http://asd-www.larc.nasa.gov/ceres/trmm/ceres_trmm.html
4. http://asd-www.larc.nasa.gov/ceres/simulator/simul.html
5. Priestly, Kopia, Lee, Mahan, Haefflin, Smith & Paden, Proc. SPIE Vol. 3221, ppf. 191, Sept. 1997